Center for Applied Autonomous Sensor Systems (AASS)
Örebro University, School of Science and Technology
SE-70182 Örebro
Sweden

OOP Lab I: Introduction

Welcome to the OOP labs! In this first lab, you will set your development environment and try to solve some simple tasks.

Part 1: DevOps – Setting up the development environment (ThinLinc, remote development)

Connecting to the remote dev machine

Instructions for Mac/Windows can be found in the following parts of this document.

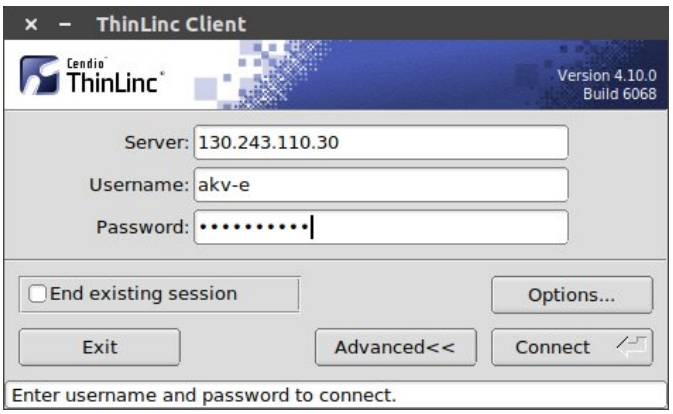
In this course, we normally use a remote development machine. The benefit of using it is that you have a separate machine (well, it is actually a whole claster with a load balancer), dedicated to the project and containing all necessary software. The machine is backed up, and is available 24/7. And you can connect to this machine from any place you like at any moment of time from any available device.

To connect to the remote dev machine, the software called ThinLinc client is used. If you are using a university machine, it should already have a ThinLinc client installed. Alternatively, if using your own computer, download and install ThinLinc Client from:

- https://www.cendio.com/thinlinc/download

Please, pay attention to what you download! You only need a CLIENT, not the server bundle!

Run ThinLinc client. When opens, connect to **130.243.110.30**, use your oru account.



Note: Click on "**Advanced**" and make sure you are connecting to **the correct server**!

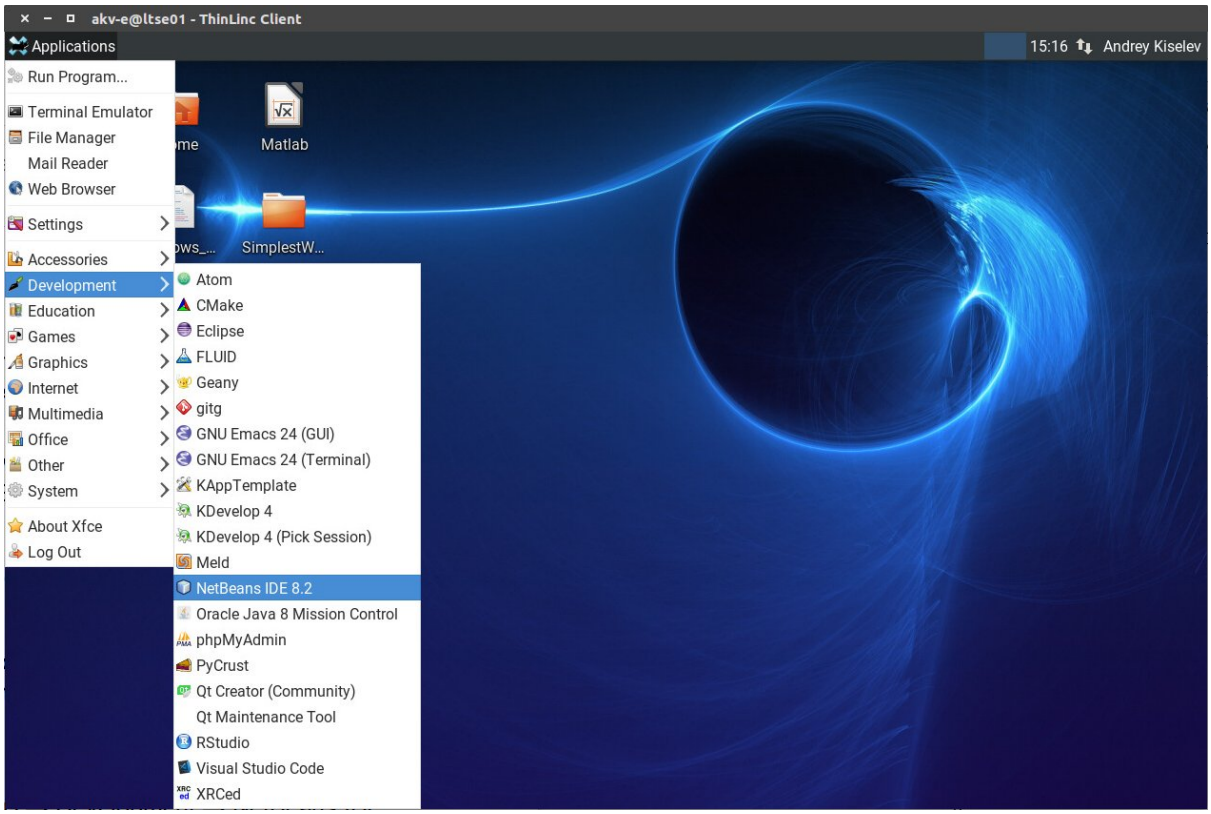
Note: you can check the options if you do not want the client to run in fullscreen mode. Also, you might want to end existing session (tick the box "**End existing session**" under the password) if something goes wrong.

The system you are now in is your development machine. It runs Ubuntu LTS with XFCE desktop environment. You can browse around, see what software is installed, access your documents, and change a desktop background.

To access your global storage, launch a script "**Windows_home**" on the desktop. Double-click on it, or use Right-Mouse-Button (RMB) and click "Execute".

Setting up IDE in ThinLinc

Now, let's start your development environment. In this course, you will use NetBeans.



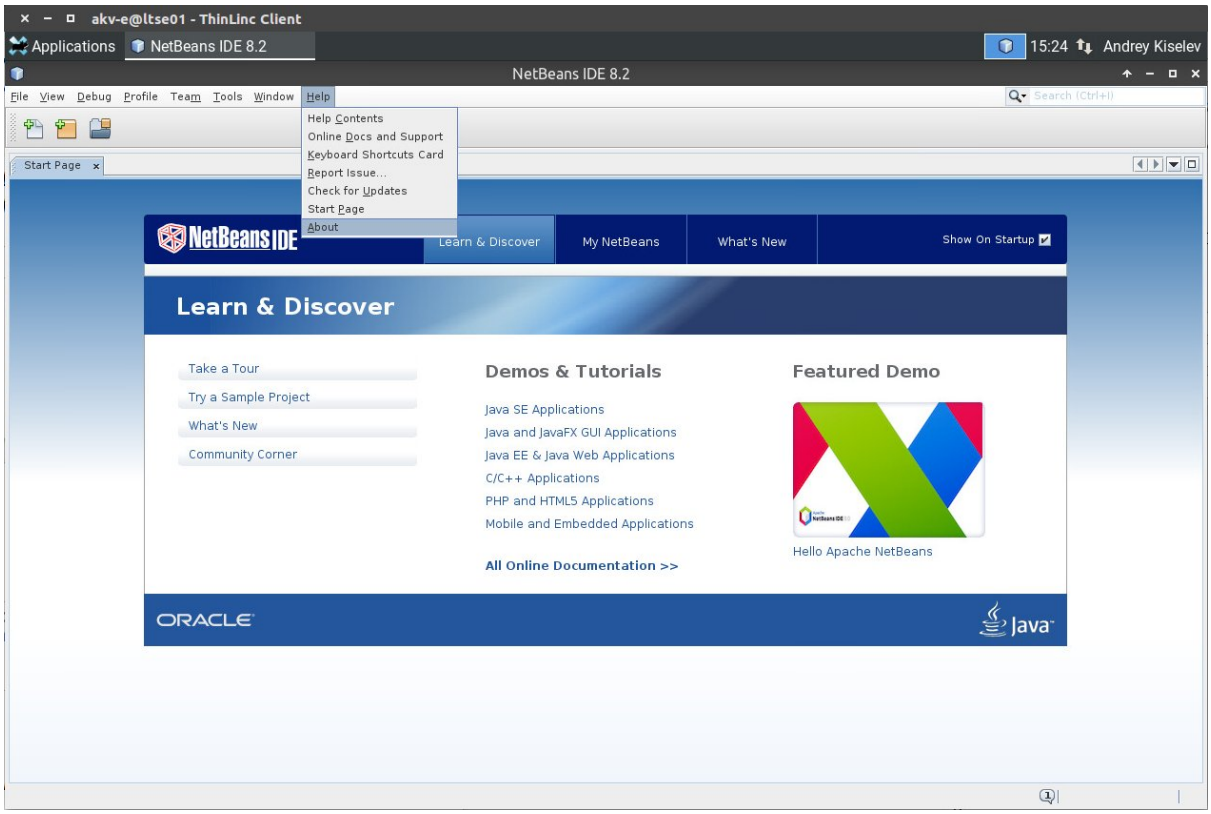
Note: *NetBeans is a free, open-source, and cross-platform development environment. It is similar to Visual Studio, which you used before. Read more about it here: https://en.wikipedia.org/wiki/NetBeans*

Note: *You are more than welcome to use any other IDE of your choice or simply go with a*

text editor (adjust lab instructions accordingly). However, teaching staff in this course is only able to help with NetBeans and might not have answers if something goes wrong with other IDEs.

***Tip:** using vi/bash throughout the whole course improves your karma by 5%!*

To start NetBeans, click on Applications (top left corner) → Development → NetBeans IDE 8.2 (you need exactly this version!). When the IDE starts, you will see something like this:



Check again that you are running NetBeans 8.2 (click Help → About to check the version)!

Now it is time to configure the NetBeans. If you run it for the first time, or cleared you installation, you need to enable some features to make NetBeans functional. Click Tools → Options to open the settings. Then the following has to be made:

- On a General tab click “Test Connection” and make sure you see a green indicator.
- Open C/C++ tab and on “Build Tools” panel click “Activate”. This this enable and set the toolset for C/C++ development. The setting windows will reload during this process and you will see the development tools. You might notice that by default NetBeans sets a GNU tool collection, which is free and open-source. You can switch to any other tools and adjust what tools to use later.

- (optional) You might want to switch to Nimbus look-and-feel. It's on "Appearance" tab, "Look and Feel" subtab.

There is nothing else to adjust right now, so click Apply and OK. This will close the Options window.

What makes the NetBeans powerful is an ability to use plugins. We will need some of them to make our life easier. To install a plugin, open Tools → Plugins. Then, click on Available Plugins tab and click on Check for Newest to update the list (see the screenshot below). The plugins that you would need in this course is as follows:

- Darcula LAF for NetBeans (optional) - will make your NetBeans dark.
- NBCndUnit - unit testing framework (optional)
- CMake Completion (optional)

New versions: NetBeans 12LTS and 12.1

The current stable version of NetBeans is 12LTS. There are not that many big differences from 8, but if using new version, you need to enable plugin store. The path to install extensions is as follows:

1. Go to Tools -> Plugins, open Settings tab and enable NetBeans 8.2 Plugin Portal.
2. Switch to Available Plugins tab and click Check for Newest
3. Select plugin C/C++ and install it, accepting the licenses and certificates.
4. Restart NetBeans
5. Go to Tools -> Options, open C/C++ tab and verify compiler toolchain in Build Tools tab.
6. Switch to Other and make sure to set Default Standard to C++14 and C11.
7. Apply, OK.

Installing NetBeans on your own computer (optional, if you do not want to use ThinLinc)

General Installation instruction for NetBeans can be found here:

https://netbeans.org/community/releases/80/cpp-setup-instructions.html

Note: *if NetBeans is not scaling well on your Windows 10 machine, try the following:*

- Open C:\Program Files\NetBeans 8.0\etc\netbeans.conf
- Change -J-Dsun.java2d.dpiaware=true to -J-Dsun.java2d.dpiaware=false

Installing NetBeans with a compilation toolchain on a lab PC (optional, if want to use NetBeans locally on the lab machine)

This part is a little bit tricky as it requires installing some libraries without admin privileges.

NetBeans 8.2 should be already installed. But the compilation toolchain needs to be installed. NetBeans itself is a text editor with some neat functionality specifically for developers like us. But it needs an external compiler to pack sources into executables. There is a large room for discussions on why one would use GNU compiler over Microsoft. Often, one might simply might not have access to MS compilation tools. Sometimes, it is required by the customer to use or not use opensource tools (which GNU compiler is). Finally, different compilers not not be exactly in line with the standard or not updated to the current standard, therefore it is up to developer which compilation toolchain to use.

The following is the installation process to get GNU compilation toolchain on a Win machine without admin priveleges:

- Download CygWin: http://cygwin.com/setup-x86_64.exe and save it somewhere.
- Make sure that you home directory is on drive M: (check in file explorer) (otherwise, change the instructions accordingly).
- Run command prompt (type cmd is search or cortana).
- Go to the folder where you saved the installer (use cd and dir commands) and run the installer with an option **`-no-admin`**.
- During the installation, set the destination folder to M:/cygwin64.
- Choose a source server in Sweden.
- In the list of packages, select **`gcc-g++`**, **`gdb`**, **`make`**, **`cmake`**, **`doxygen`**. You might also look for tools to compile clean C, assembler, or fortran, if you will.
- Finish the installation.

After that, there are some things to tweak in NetrBeans to let it know where the compilation toolchain is installed:

- Open NetBeans and go to Tools – Options.
- In C++ tab, click on Add under Tool Collection.
- In the tool adding window, choose M:/cygwin64/bin as a Base Directory.
- If all previous step were correct, NetBeans should automatically find the rest of configuration parameters.

Part 2: Working with Git repositories

Git on the Web

During the whole course, we will be using git for all code development.

From Wikipedia:

> *Git (/gɪt/[7]) is a version control system for tracking changes in computer files and coordinating work on those files among multiple people. It is primarily used for source code management in software development,[8] but it can be used to keep track of changes in any set of files. As a distributed revision control system it is aimed at speed,[9] data integrity,[10] and support for distributed, non-linear workflows.[11]*
>
> *Git was created by Linus Torvalds in 2005 for development of the Linux kernel, with other kernel developers contributing to its initial development.[12] Its current maintainer since 2005 is Junio Hamano.*

More information on git is available at: https://git-scm.com/

As with most other distributed version control systems, and unlike most client–server systems, every Git directory on every computer is a full-fledged repository with complete history and full version tracking abilities, independent of network access or a central server.

Git is widely used as a command line tool, however, graphical tools also exist. In this course, we will use NetBeans functionality to work with repositories.

Before we jump right into messing things up, please read a basic git tutorial at https://git-scm.com/doc. You will need to make sure that you understand the content of basic concepts and life cycle.

Our Git server is located at https://git-e.oru.se. Use your student login/password to sign in. As soon as you are logged in, search around. Look at your profile, check settings and preferences. ***You might be interested in creating an SSH key to push and pull code using SSH. Read about it on your profile page.*** You will also make teachers' lifes much easier if you will use your real name (or at least one on the blackboard) in the git repository. **Thank you!**

On git server, create a new project, giving it some name (e.g. "OOP-Lab-I"). You can make your project private for now.

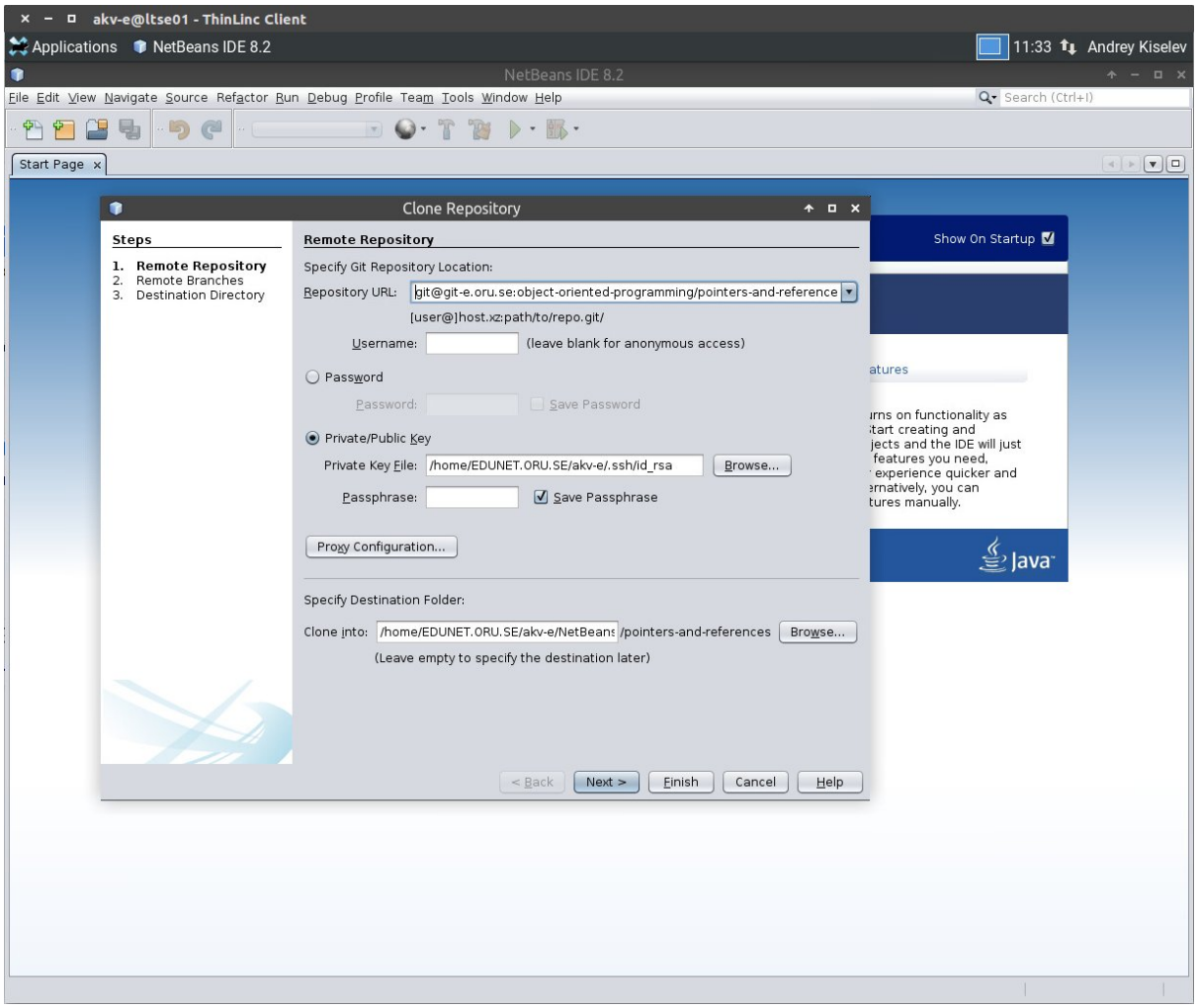
When you feel familiar enough with the things around, go to group and exprore available repositories. You will be able to see all public and internal repos. You are looking for one, called "Pointers and References". That repository contains some important tasks for the first lab.

Git in NetBeans

The following procedure shows how to clone and work with repositories in NetBeans:

- From the project web page, copy the repository address. You need the one with SSH access, if you created a SSH key before.
- Open NetBeans (NB)
- In NB, click Team - Git – Clone

- Put the repo address and the private key.



- Choose directory to clone to, agree with default options and folow the cloning process.
- When the repository is downloaded, the NB should offer you open any existing project. Agree with that.
- Ta-da!!!

Now, you have an exact copy of the repository on your machine. Although you can do anything and then commit to the main branch, normally, developers create own branches to make changes and then ask the owner of the repo to make a merge. That is what you will do.

- Make sure your active project is the one you just cloned from the repo.

- Click Team - Branch/Tag - Create branch.
- Name the branch using your unique login name to make sure there are no branches with the same name.
- Double click on the the main.cpp file to open it. Then, Clean and Build (Shift + F11) and Run (F6). Should see “RUN FINISHED; exit value 0; real time: 0ms; user: 0ms; system: 0ms” in the output console.

Now it is time to push your branch on the server (although it is practicaly identical to the master):

- Click Team - Commit and put some meaningful message to the "Commit Message" field. A wonderful guide on writing commit messages is available here: https://chris.beams.io/posts/git-commit/
- Now, click Team - Remote - Push to upload all you changes on the server. In the window, check all branches and agree to set your local new branch to track remote.

Finally, it is time to check the repo on the server. Go you git-e server and reload the project page. In the branches tag you should now see both, the master and your new branch.

If your repo contains some meaningful code, the final step is to merge changes from your repository to the master. On the repository web page, click "merge request" on you branch to make it merged into the master. Again, you should be fine with the default options, just doublecheck what branches you are trying to merge from and to. At this step, TAs will decline your merge requests, so you keep working on the local branch.

Part 3: Pointers and References

The tasks are replicated in the repository readme.md file!
C++ is very similar to pure C. Use your knowledge and intuition to solve the following task.
Upon completing it, Clean, Build, Run (make sure it actually runs!), Commit, and Push to the server!

Task 3.1: Passing Function Arguments

Before you dig into this task, refresh your knowledge on passing function arguments by value or by reference!

Write a bubble sort function in C++, which takes an array of integers as an argument and returns a sorted version of the array. Make two versions of the function: passing arguments by reference using pointer and passing arguments by reference using C++ reference. Discuss with your BFFs proc and cons of every method.